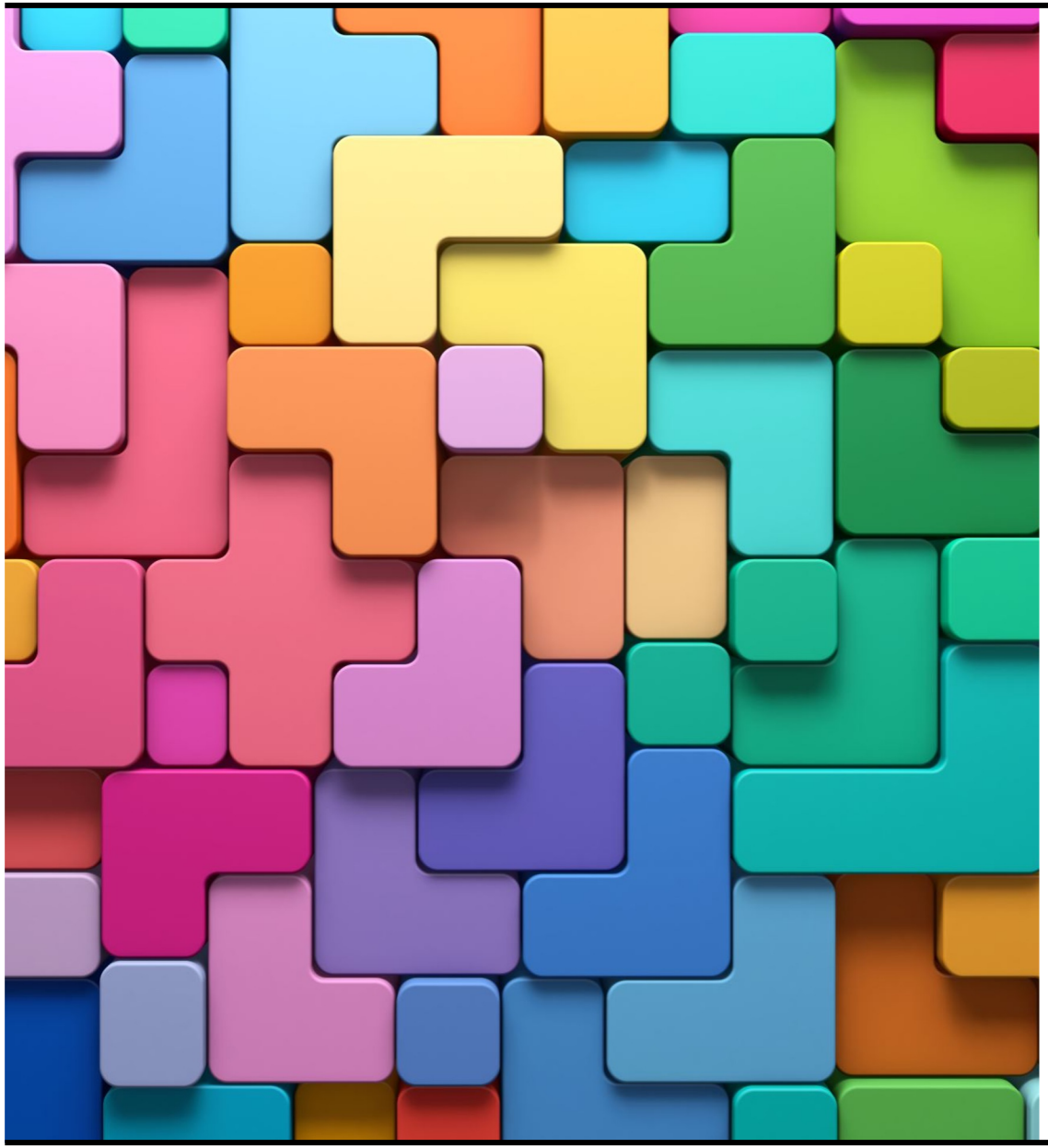


IMDB Sentiment Analysis

William Owens

Sentiment Analysis

Natural Language Processing (NLP) task that involves classifying opinions expressed in text.

We will be performing binary sentiment analysis on the IMDB movie review dataset.

The two classes will be positive sentiment and negative sentiment.

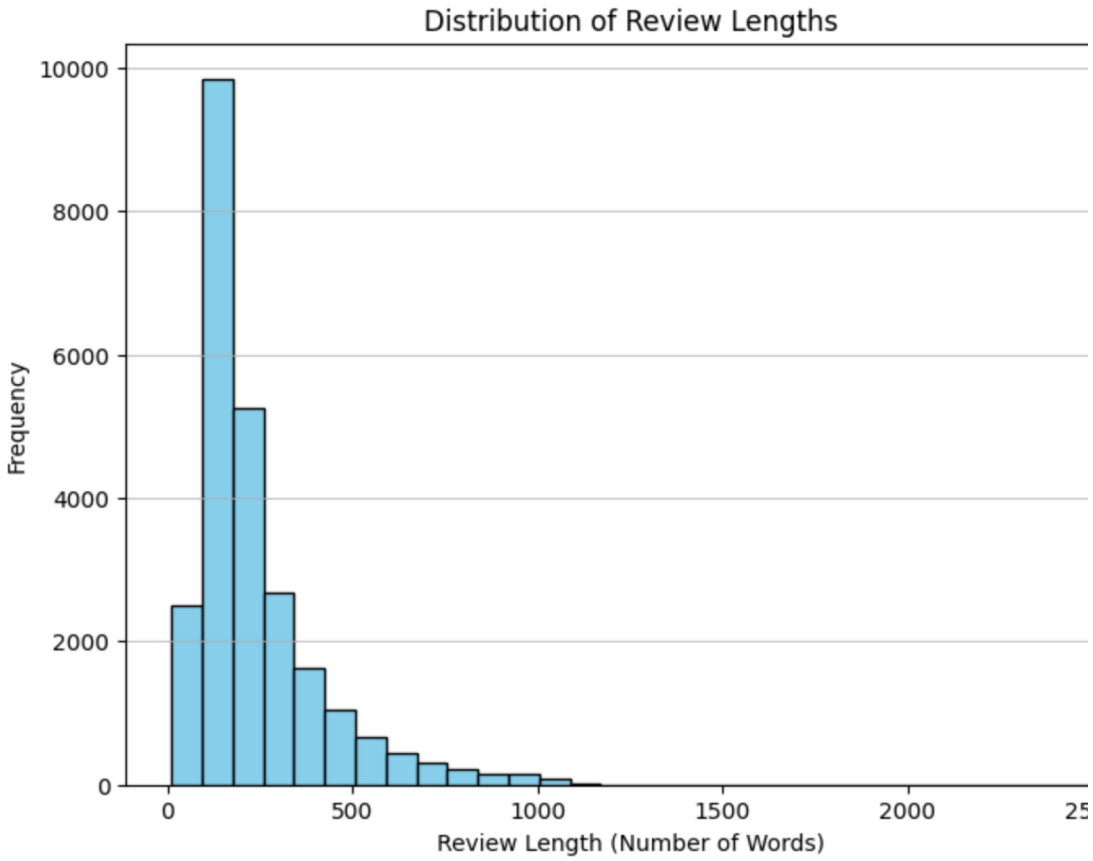
IMDB Dataset

The IMDB datasets contains 50,000 movie reviews. Initially, there are 25,000 reviews assigned to the training set and 25,000 reviews assigned to the test set.

Each review is labeled as either positive or negative in terms of the sentiment expressed in the review.

The reviews in the IMDB dataset loaded from Keras have been converted into sequences of integers, where each integer represents a specific word in a dictionary.

Data Exploration



The distribution of review lengths showed that most of the reviews are around 200-250 words long.

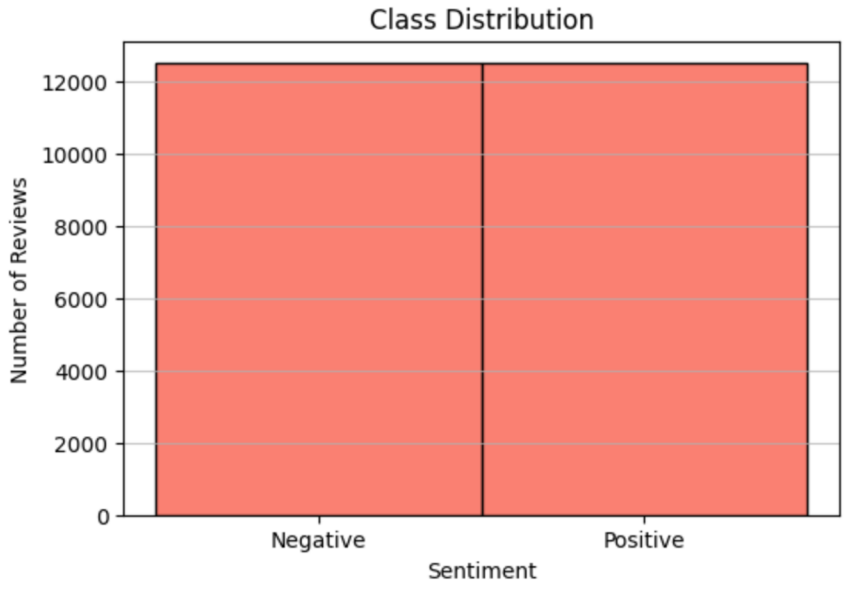
We observed a right-skewed distribution with some of the longest reviews being around 1,000 words long.

We will use this information to determine our maximum sequence length for the neural network models.

Data Exploration

An examination of the class distribution showed that there are an equal number of positive and negative reviews in the dataset.

When we create our validation set, we will need to maintain this class balance.

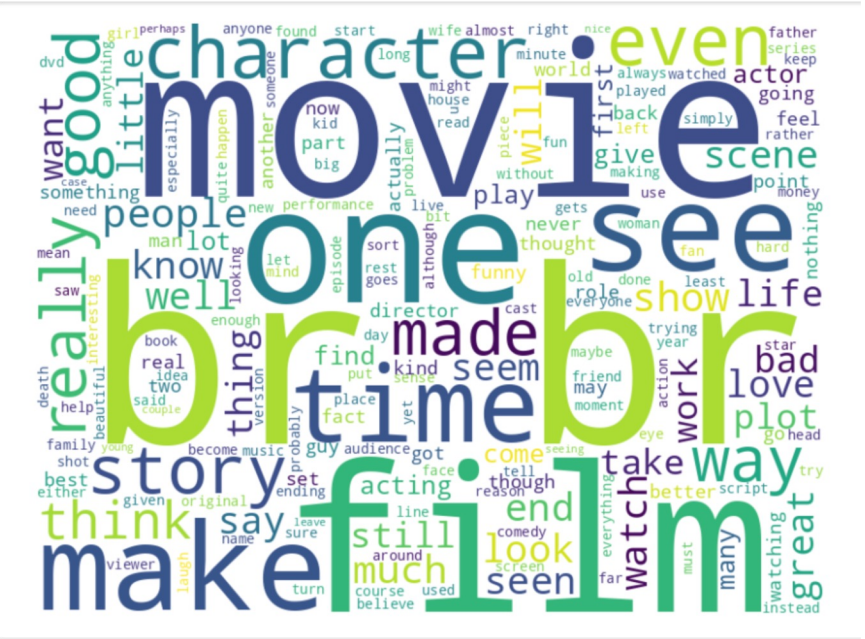


Data Exploration

The size of each word in a word cloud indicates its frequency or importance in the dataset.

Based on the word cloud of the entire dataset, the largest, and therefore most frequent, words appear to be neutral in tone.

The word “good” is slightly larger than the word “bad” in the word cloud. Perhaps positive sentiment will be easier to detect.

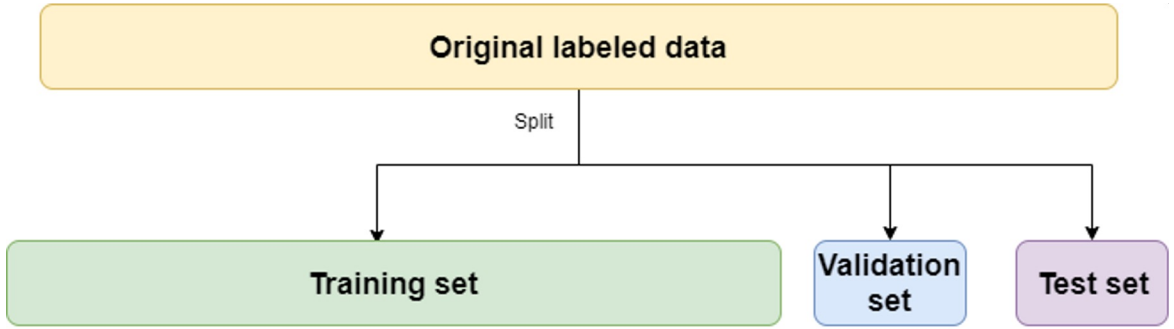


Splitting the Data

The training set has 25,000 reviews.

We split the test dataset (25,000 reviews) into a validation set and test set with 12,500 reviews each. We used a stratified split to maintain the class balance.

The training, validation, and test sets were randomly shuffled.



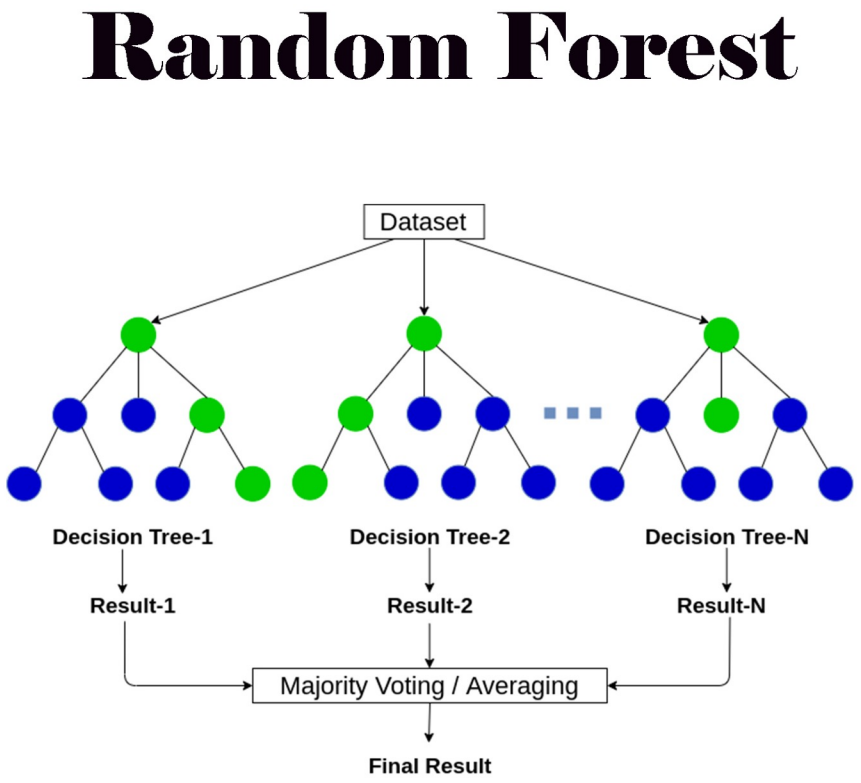
Models

We will use four different types of models for this sentiment analysis task.

We will use two machine learning models: a random forest and support vector machine.

We will use two deep learning models: a convolutional neural network and a long short-term memory neural network.

Random Forest



Random forest is an ensemble learning method. Multiple decision trees are created, and their predictions are aggregated to obtain the “wisdom of the crowd”.

Each tree in the forest is trained with a different random subset of the data and considers different random subsets of features (words) when making its decisions.

This leads to a diverse set of decision tree models that produce an aggregated prediction, which is more resistant to overfitting.

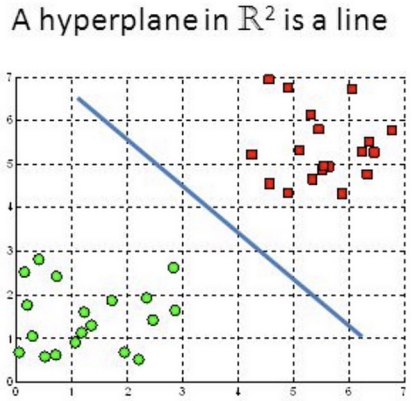
Support Vector Machine

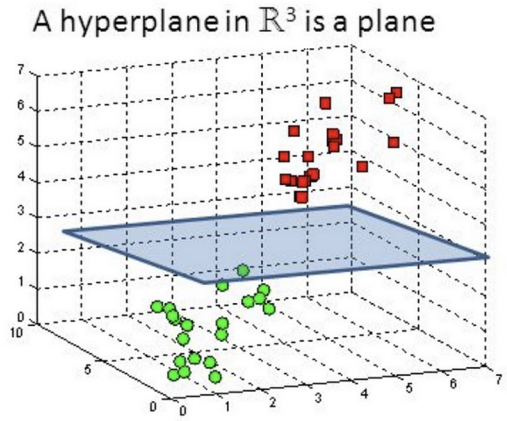
SVM finds the hyperplane that best divides the dataset into its respective classes.

Data points closest to the hyperplane are called support vectors and they influence the position and orientation of the hyperplane.

The kernel trick allows us to map the input features into a high-dimensional space without performing laborious computations.

Hyperparameters C and gamma are important for preventing overfitting.





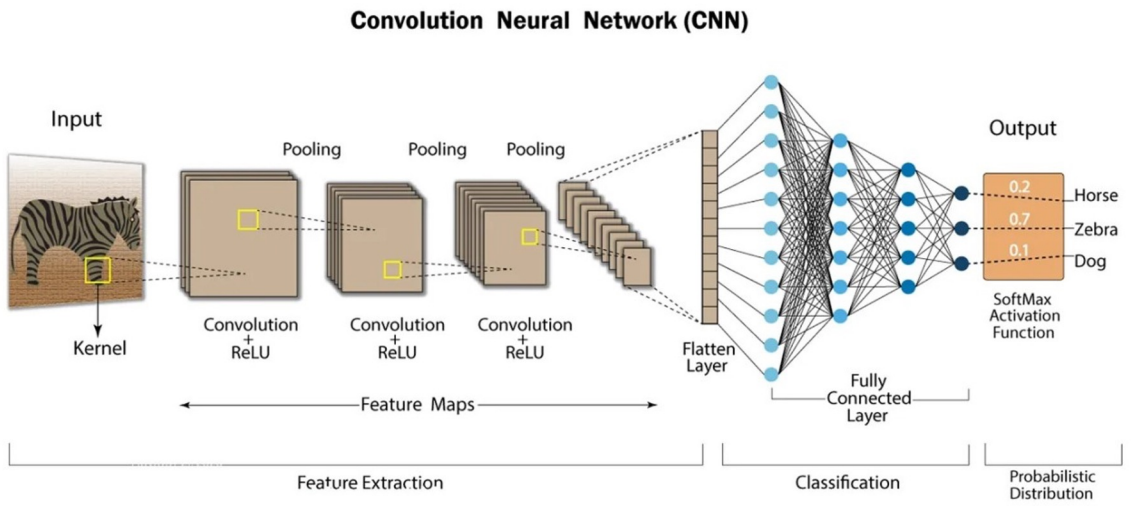
Convolutional Neural Network

Convolutional neural networks are deep learning models that are commonly used for image processing.

They include convolutional layers that filter the input to extract features, pooling layers that reduce the dimensionality of the feature map produced by the convolutional operations, and fully connected layers that process the feature map for classification.

Residual connections are an innovation that has enabled deeper CNN architectures. They allow for the direct flow of information across different layers of the network, reducing the vanishing gradient problem.

In an NLP context, CNNs can learn the local and hierarchical relationships between words in the texts.



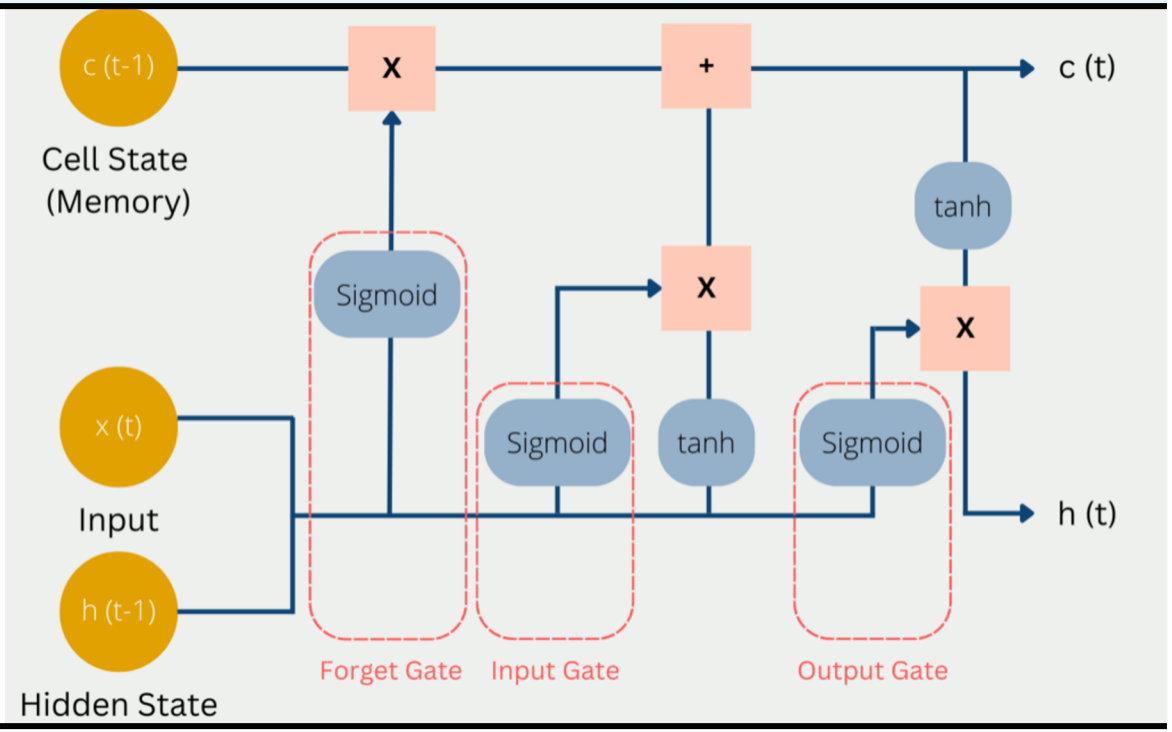
Long Short-Term Memory

LSTM neural networks are a type of recurrent neural network that help to address the vanishing gradient problem and allow models to learn long-term dependencies.

They are well-suited for sequential data such as text data or time series data.

The input, forget, and output gates collectively determine which information gets passed through the network, which gets retained in the cell state, and which gets forgotten.

We will explore different architectural configurations of the LSTM, including variations with skip connections, residual connections, bidirectional LSTM layers, an encoder-decoder approach, and a multi-head attention mechanism.



Processing for ML Models

We use count vectorization to process the data splits for use with our machine learning models.

A vocabulary-length vector is created for each review with the vector values corresponding to the count of the various words in the review.

We also use term-frequency-inverse-document-frequency vectorization.

This can capture the importance of the word within the review.

Truncated single value decomposition (SVD) is applied to the BoW and TF-IDF data to reduce dimensionality specifically for the SVM modeling.

$$w_{x,y} = tf_{x,y} \times \log\left(\frac{N}{df_x}\right)$$

TF-IDF

Term x within document y

$tf_{x,y}$ = frequency of x in y

df_x = number of documents containing x

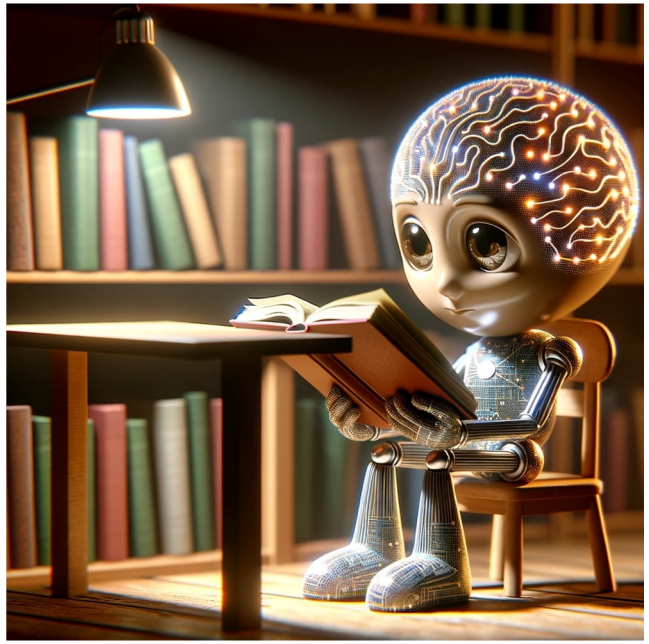
N = total number of documents

Processing for DL Models

The dataset is initially processed to be frequency-ranked and integer-indexed.

We pad the data with a maximum sequence length of 250.

We will allow the models to learn their own embeddings for the reviews.



RF Results

The random forest model had the highest accuracy on the data processed with BoW (left) compared to the data processed with TF-IDF (right).

	Score
Accuracy	0.849360
Precision	0.849752
Recall	0.848800
F1 Score	0.849276
ROC-AUC	0.849360

	Score
Accuracy	0.843200
Precision	0.855015
Recall	0.826560
F1 Score	0.840547
ROC-AUC	0.843200

SVM Results

The support vector machine model had the highest accuracy on the data processed with TF-IDF (right), compared to the data processed with BoW (left).

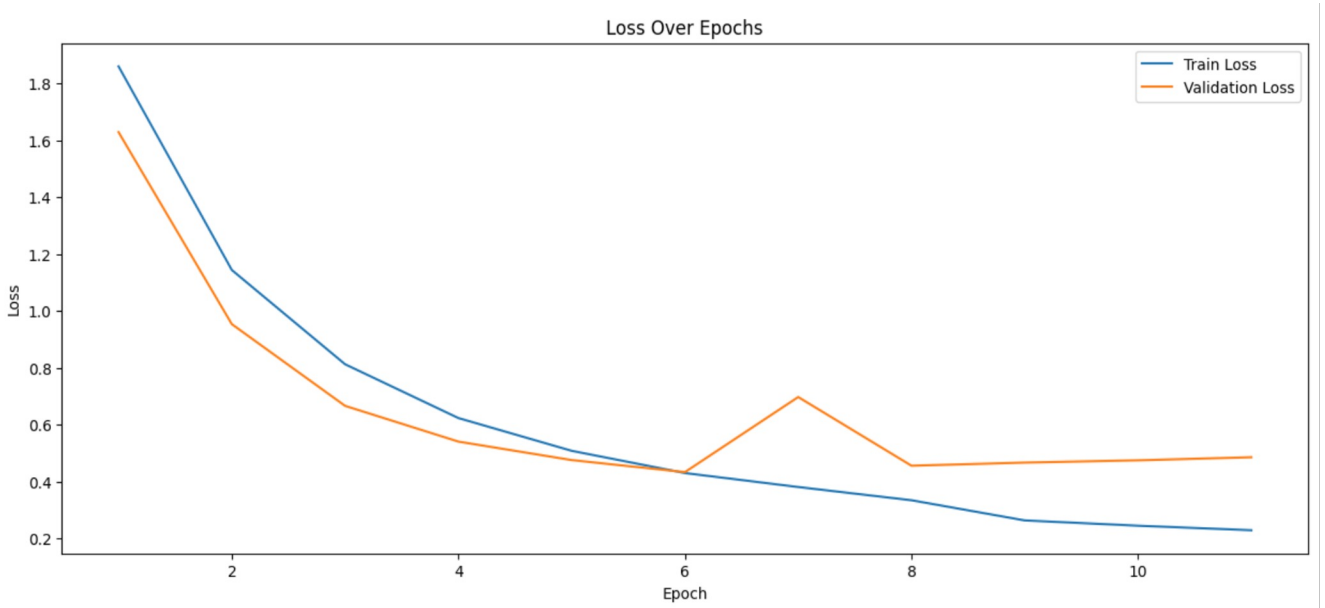
	Score
Accuracy	0.783120
Precision	0.775409
Recall	0.797120
F1 Score	0.786114
ROC-AUC	0.783120

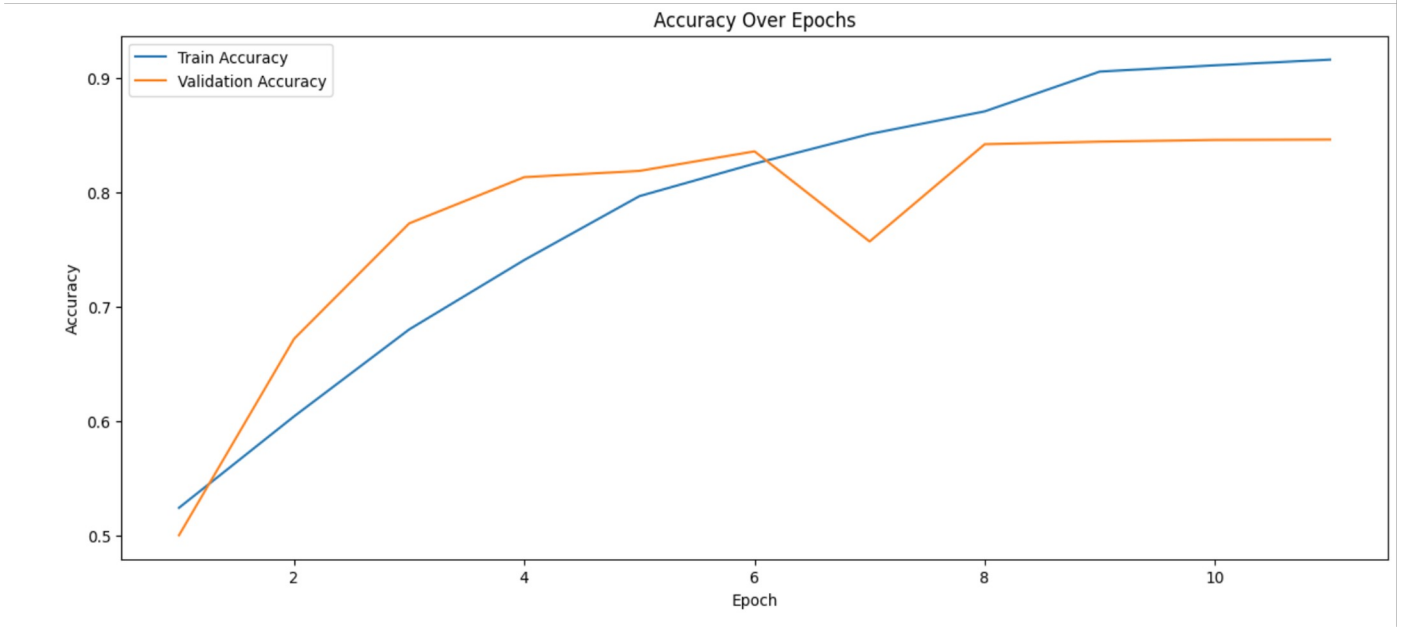
	Score
Accuracy	0.850880
Precision	0.848095
Recall	0.854880
F1 Score	0.851474
ROC-AUC	0.850880

CNN Results

CNN was trained with 2 residual blocks with two convolutions each, L2 regularization of 0.01, batch normalization, and ReLU activation function.

	Score
Accuracy	0.836080
Precision	0.847938
Recall	0.819040
F1 Score	0.833238
ROC-AUC	0.916453



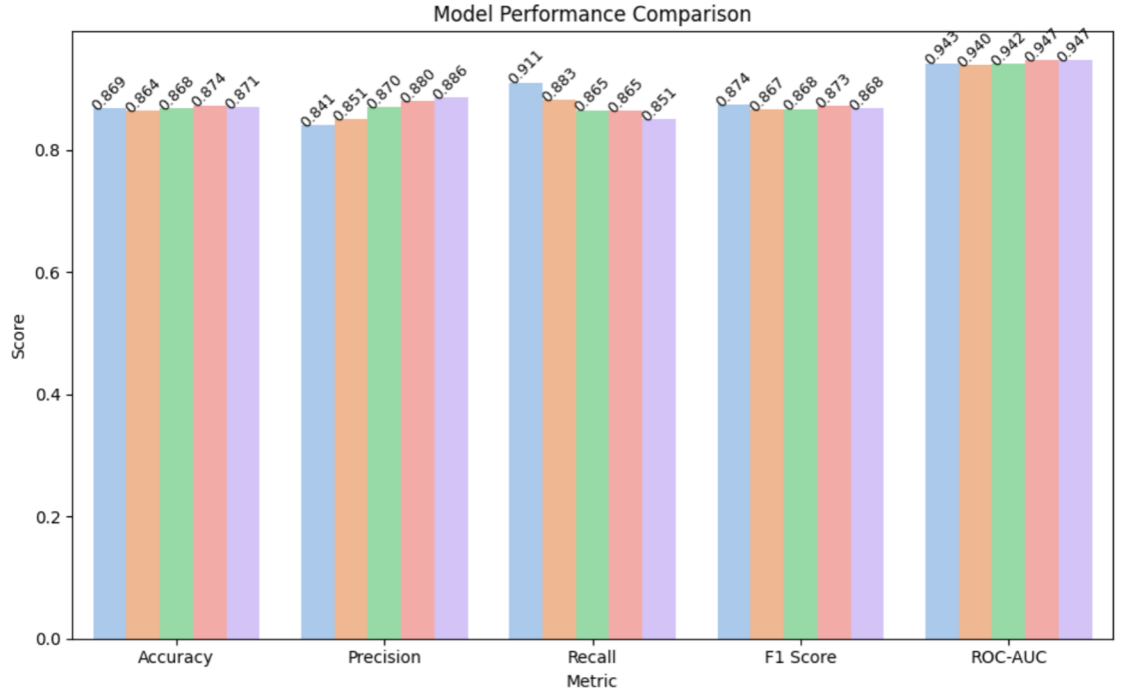


LSTM Results

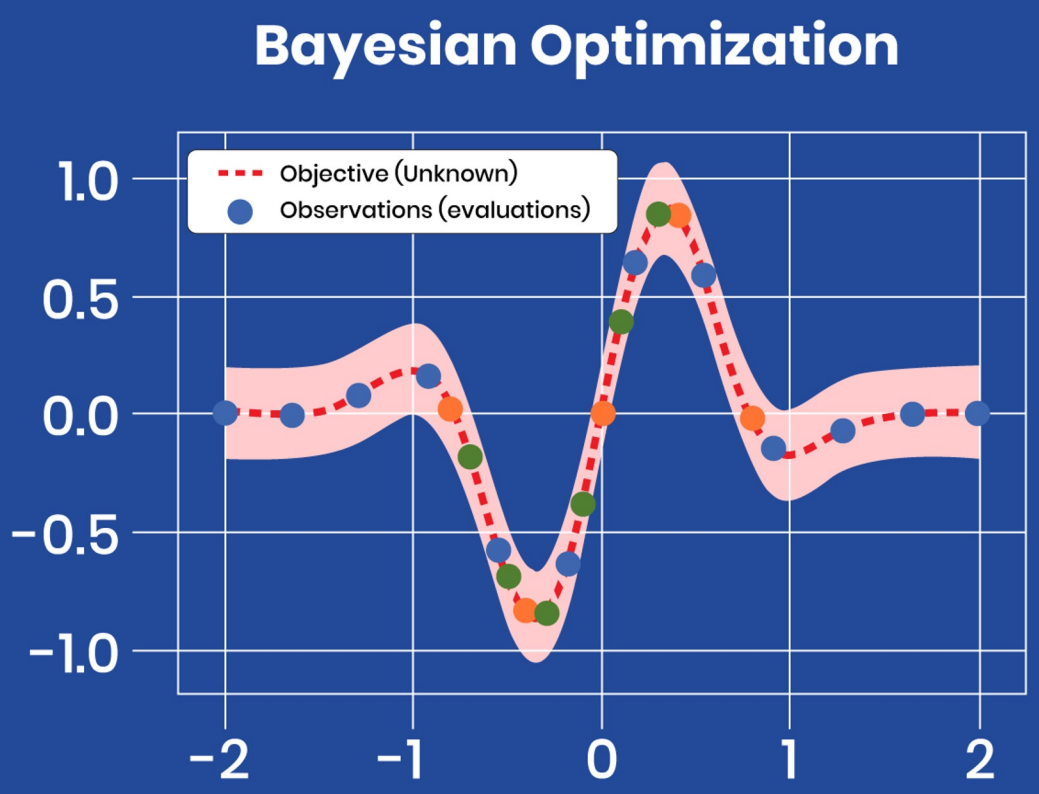
LSTM was trained with five different design configurations. We created models that applied either: skip connections, residual connections, bidirectionality, an encoder-decoder structure, or a multi-head attention mechanism.

The models with the encoder-decoder structure or multi-head attention mechanism performed the best on accuracy and precision. The skip connection and residual connections models had the highest recall.

Multiple runs showed that the skip connection model was quite volatile. All models overfit very quickly.



Hyperparameter Tuning



We tuned the four types of models, using the BoW random forest model, the TF-IDF support vector machine, the convolutional neural network, and the attention mechanism LSTM (but with bidirectional LSTM layers).

We used the Optuna library to perform Bayesian optimization. Educated guess is made about next iteration's parameter set based on prior results and the probabilistic model of the objective function.

Model Comparison

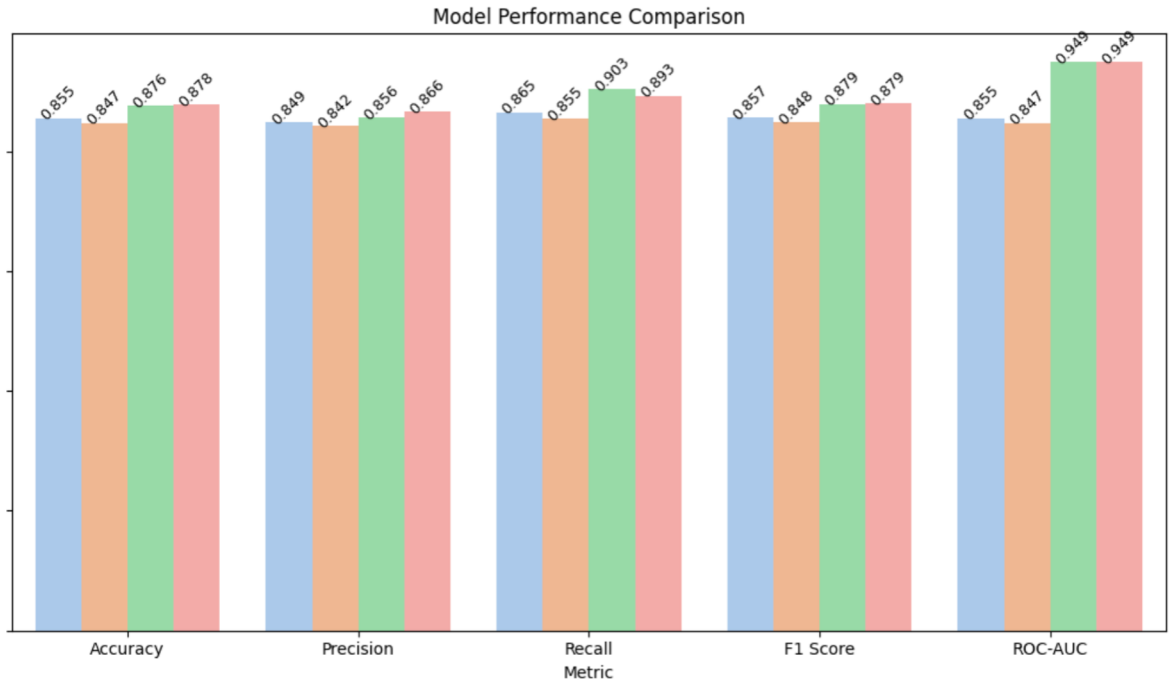
We evaluated the four tuned models on the test set.

Ultimately, the LSTM model had the highest accuracy and precision.

The CNN model had the highest recall.

The CNN and LSTM scored equally well on the F1 score.

The DL models outperformed the ML models on all metrics, but not by a large amount.



Inferencing

We used both models to predict the sentiment of a new review.

The new review was: “This movie was fantastic! I loved it.”

We preprocessed the new review properly for each of the models.

All four models correctly predicted positive sentiment.

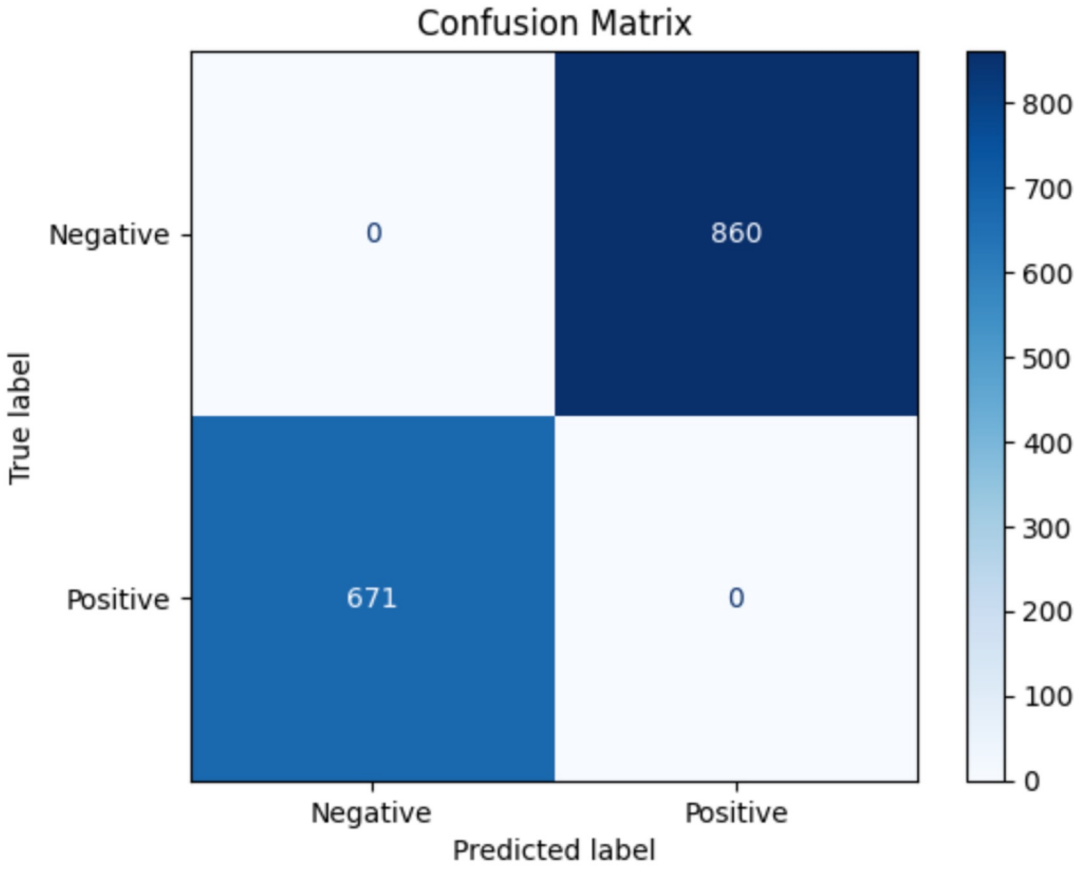
```
RF Prediction: Positive Sentiment

SVM Prediction: Positive Sentiment

CNN Prediction: Positive Sentiment

LSTM Prediction: Positive Sentiment
```

Misclassification Analysis



The LSTM model was selected as the best model to use for our subsequent misclassification analysis.

The confusion matrix of the misclassified reviews shows that the model had more false positives than false negatives. It tends to overpredict positive sentiment.

Raising the threshold of the classifier may improve classification precision.

Misclassification Analysis

Topic modeling on the misclassified reviews showed that the word “good” is present in all the topic representations. This indicates that the model may be predisposed to predict positive sentiment if it sees the word “good”, even if the review overall reflects negative sentiment.

```
Topic 0:
film like br man films movie story just people good


Topic 1:
br film movie life good mother time characters scenes does


Topic 2:
movie film like br good just people time funny make


Topic 3:
br movie film like just good really time don people


Topic 4:
movie film like good bad time just better story think
```

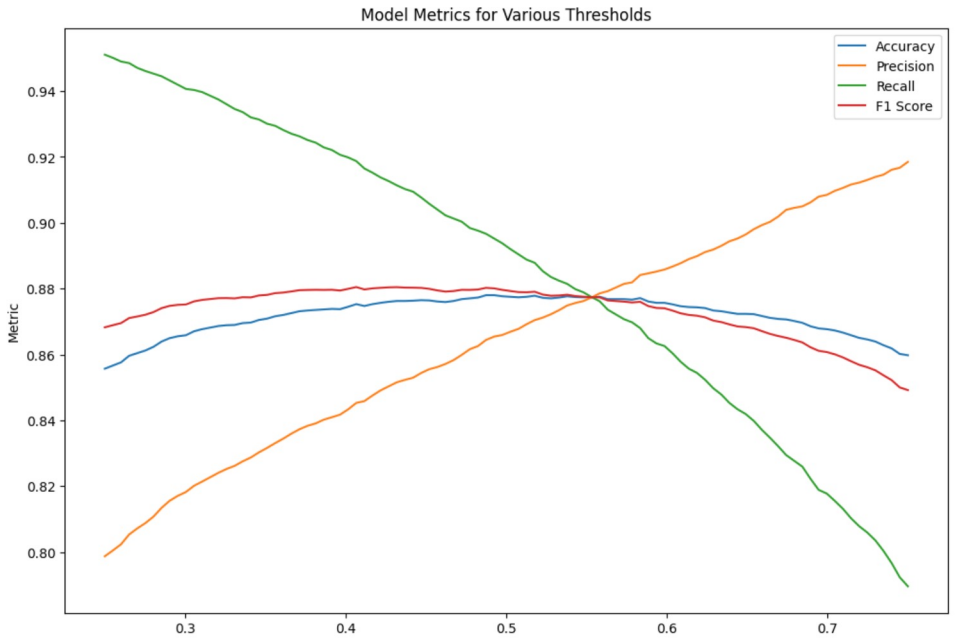
Threshold Analysis

We created some negative reviews that the model predicted as positive with a probability of 0.501 - 0.520 of belonging to the positive class.

We decided to explore altering the decision threshold for classifying the reviews. The initial threshold was 0.5.

A plot of varying threshold scores and the model metrics was created. This showed that increasing the threshold would result in greater precision without costing much in the way of accuracy.

This is a band-aid solution, but it will allow us to improve model performance while we work out the architectural kinks.



Threshold Analysis

We decided to set the new threshold based on maximizing precision while maintaining the lowest acceptable recall. We settled on a recall score of 0.85 as the lowest acceptable recall. The optimal threshold based on this precision-recall trade-off was 0.6285715699195862.

We used this as the new threshold and the negative reviews we wrote earlier were now being correctly classified as negative by the model.

Example: “The movie was not what I was expecting. It was a bit too long. The characters were lame. I won’t ever see this movie again. I don’t recommend it to anyone.”

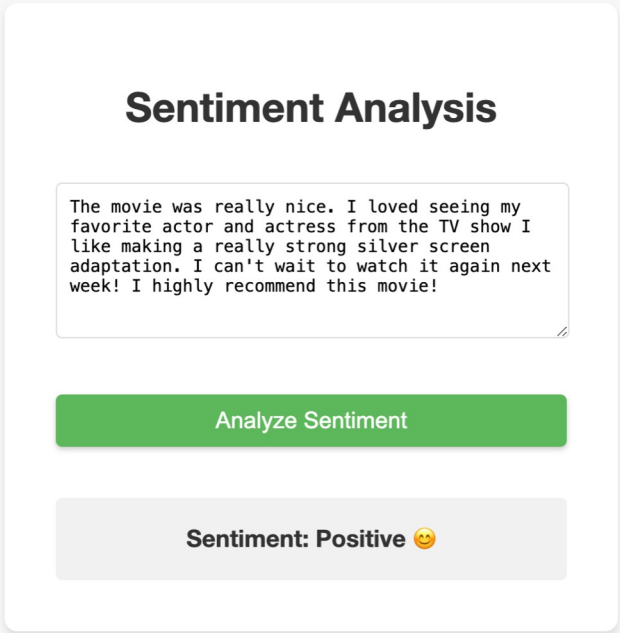
Our updated model predicted a raw probability of 0.58. If we had kept the original decision threshold, the model would have predicted positive sentiment, which is incorrect.

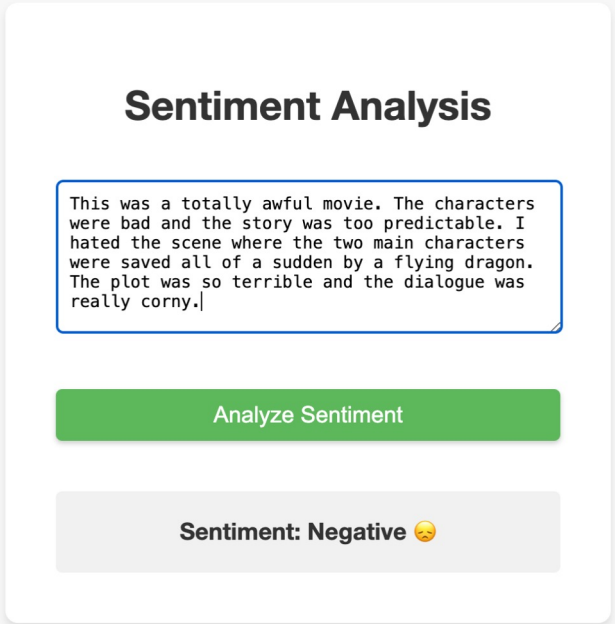
```
Raw Prediction: [[0.58328456]]
'Negative'
```

App Prototyping

We used Flask to create an app prototype.

Users can enter a movie review and the app predicts positive or negative sentiment.



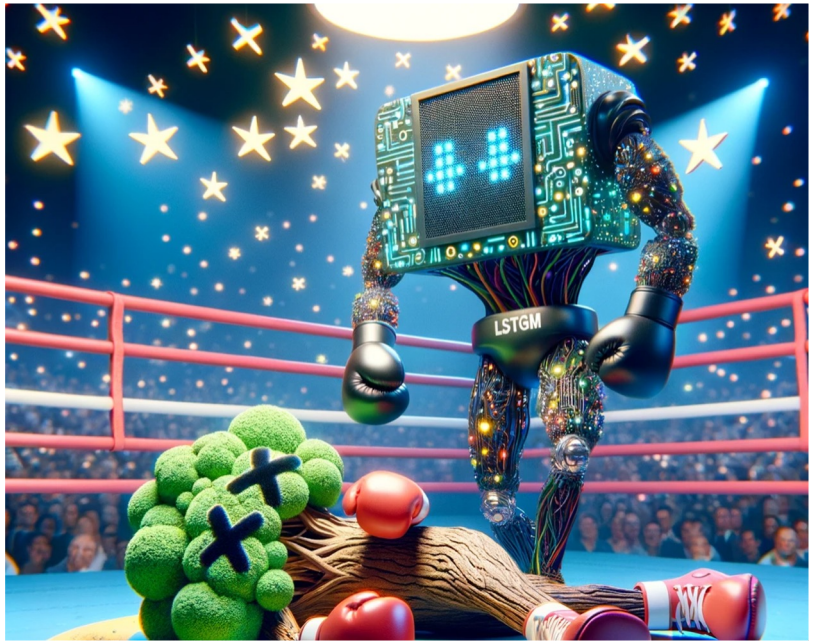


Conclusions

Ultimately, the LSTM model performed the best out of all four models.

The deep learning models outperformed the machine learning models, although not by a large amount.

This further validates the superiority of LSTM models for text-related tasks.



Future Directions

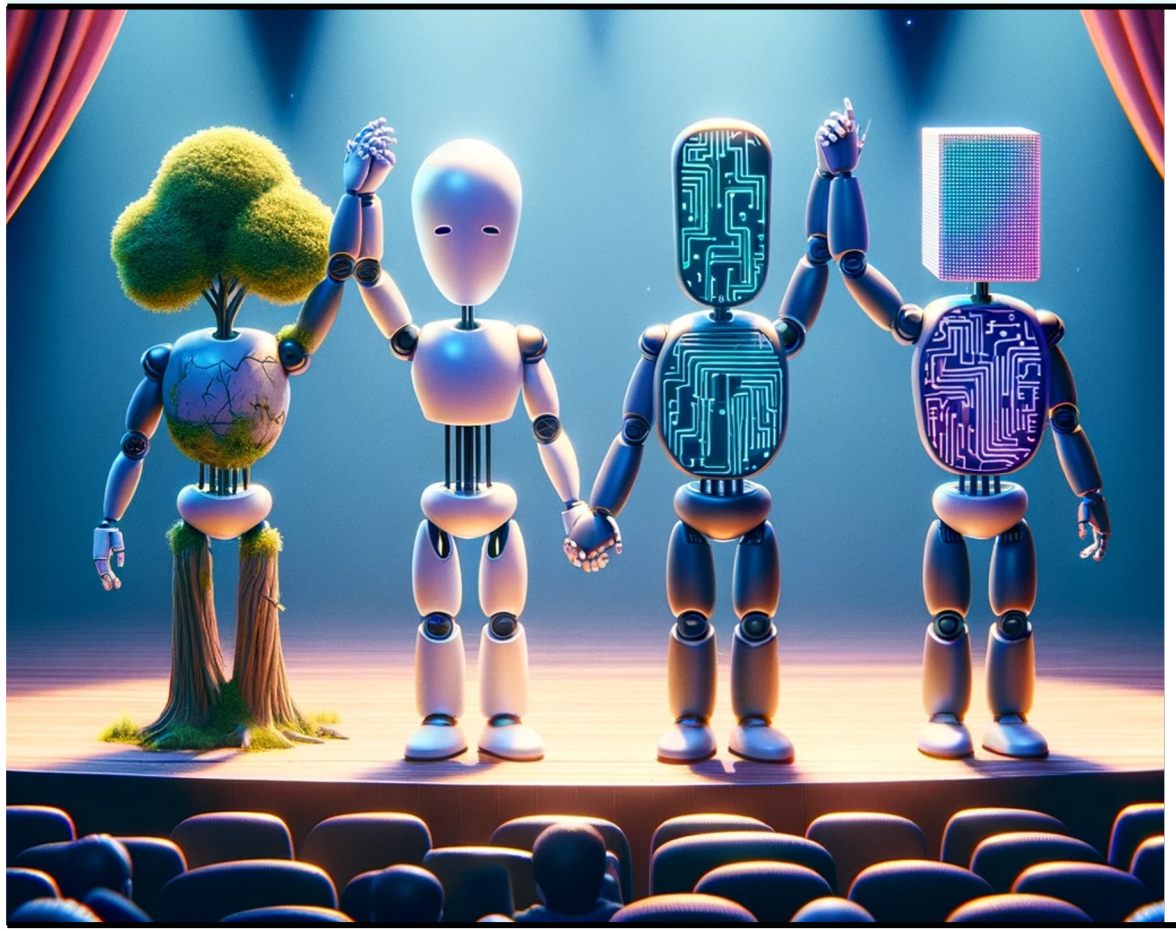
Future research may explore different preprocessing procedures for the various models.

Perhaps pretrained embeddings such as Word2Vec, Doc2Vec, GloVe, or RoBERTa-extracted embeddings may result in greater performance.

Different architectures for the LSTM model should be explored. Perhaps combining all the different design techniques would have resulted in greater performance.

Interesting approaches could involve scaling the embeddings using the weights of another classifier or feature importance scores from a tree-based classifier.

Thank You!



Any questions?